Practical 2: Travelling Salesman Problem by GA

Computational Intelligence Assessment: L4 EC2

Zijin Hong
2020101911
Mathematics and Applied Mathematics
Jinan University - JBJI
`hongzijin@stu2020.jnu.edu.cn`

1 Introduction

This is a report of practical project[1]: Travelling Salesman Problem by GA, the catalogue including:

- Part A, the compulsory task's GA result and the required questions answering.
- Part B, GA execution for optional challenge for 70 cities TSP.

2 Part A: Compulsory Task

2.1 Q1: "How will you encode the solution?"

For the Traveling Salesman Problem (TSP) provided, the solution (i.e., a route that visits each city exactly once and returns to the starting city) can be encoded as a permutation of city indices.

For example, if there are 8 cities, a possible solution might be represented as: [3, 1, 7, 6, 5, 4, 2, 0]. This would correspond to the route: $D \to B \to H \to G \to F \to E \to C \to A \to D$.

Using this encoding method, the route is represented as a list of city indices in the order they are visited. And operations like crossover (recombination) and mutation can be easily applied to generate new routes (solutions) for the next generation in the Genetic Algorithm.

2.2 Q2: "How will you estimate the solution?"

To estimate the solution for the TSP problem, one effective way is using greedy methods, one can use the "Nearest Neighbor" heuristic. Here's how the Nearest Neighbor method's algorithm works:

Algorithm 1 Nearest Neighbor for TSP

```
1: Initialize: Start at a randomly selected city.
2: while there are unvisited cities do
3:     Find the closest unvisited city based on the distance matrix.
4:     Move to that city.
5: Return to the starting city.
```

Using this algorithm in our scenario, by selecting A as the start city, we can obtain a target route: $A \to D \to H \to F \to C \to B \to G \to E \to A$. The corresponding total length of this route is 36. This method provides a good, though not guaranteed optimal, solution quickly.

[1]The code is available in `https://github.com/Rcrossmeister/CI/tree/main/P2`

2.3 Q3: "How will you set the fitness function?"

For the TSP, the most natural way to define the fitness of a solution is inversely related to the total distance of the route. For a route r, the fitness $f(r)$ is defined as the inverse of the total distance of the route:

$$f(r) = \frac{1}{\text{total distance of route } r} \tag{1}$$

We want routes with shorter distances to have higher fitness values, so they are more likely to be selected during the selection phase of the genetic algorithm. Then this fitness function ensures that the genetic algorithm will search for solutions that minimize the total distance of the TSP route.

2.4 Q4: "How will you employ evolutionary operators to generate new offspring, what about the parameter setting? Are they effective?"

Employing Evolutionary Operators:

Selection: Routes are selected based on fitness. Methods include Roulette Wheel (probability proportional to fitness) and Tournament selection (best among a few randomly chosen).

Crossover: Combines two routes. Techniques include Ordered Crossover (OX) and Partially Mapped Crossover (PMX).

Mutation: Introduces variability in a route. Techniques are Swap (two cities are exchanged) and Inversion (a segment is reversed).

Parameter Setting:

1. **Population Size:** 50 to 200
2. **Crossover Rate:** 0.7 to 0.9
3. **Mutation Rate:** 0.01 to 0.05
4. **Number of Generations:** 100 to 1000 (based on convergence)
5. **Elitism:** Top 1%-5% routes for the next generation

Effectiveness: Operators like OX and PMX are typically more effective for TSP. Parameter tuning is necessary for the specific problem instance. Generally, these provide near-optimal solutions quickly but do not guarantee a global optimum.

2.5 Q5: "How will you set the terminal criterion? Does it perform effectively?"

For the TSP scenario using a genetic algorithm:

Terminal Criteria:

1. **Maximum Generations:** Limit the number of iterations.
2. **Convergence:** Stop if there's minimal improvement over several generations.
3. **Time Constraint:** Halt the algorithm after a predefined time.
4. **Known Optimum:** Terminate once a specific solution value is achieved.
5. **Population Diversity:** End if variation in the population drops below a threshold.

Effectiveness: The best terminal criterion often depends on the problem's specifics. For instance, if computational time is a concern, a time constraint might be most effective. However, for solution quality, monitoring convergence can be ideal. It's often beneficial to combine multiple criteria for a balanced approach.

2.6 Q6: "What is your final results?"

Using the GA solve our scenario, we can obtain a solution[2] of optimal route: $G \rightarrow B \rightarrow C \rightarrow F \rightarrow H \rightarrow E \rightarrow A \rightarrow D$. The corresponding total length of this route is 24, which is the minimum of the all the total route lengths.

3 Part B: Optional Challenge

3.1 Distance Matrix

To address the TSP for these 70 cities, we'll first need to compute a distance matrix based on the city coordinates. Naturally, we use the Euclidean distance formula to determine the distance between two cities:

$$D(i,j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \tag{2}$$

Where $D(i,j)$ represent the distance between the city i and city j. We can compute the distance mutually and $D(i,j)$ is exactly the (i,j) element in the distance matrix.

3.2 Hyperparameters

We setting the parameters of our GA:

1. **Population Size:** 100
2. **Crossover Rate:** 0.9
3. **Mutation Rate:** 0.01
4. **Number of Generations:** 5000
5. **Elitism:** Top 5% routes for the next generation

3.3 Results

After setting up, we then do the execution of GA, we have the convergence process with generation increasing:

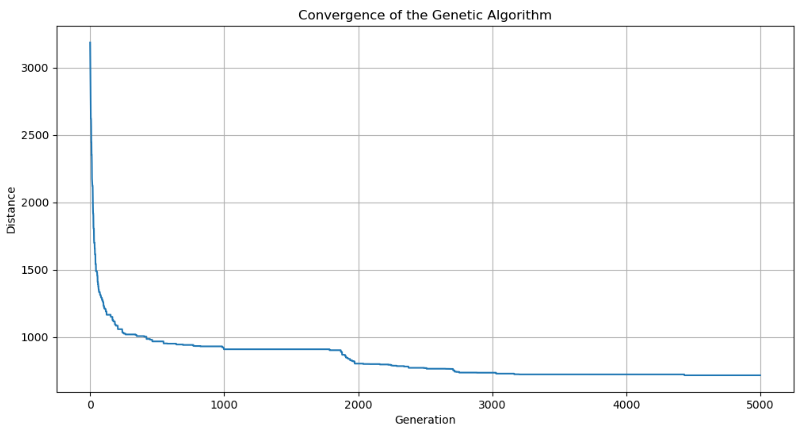

Figure 1: Convergence process

We can see the GA successfully converge after 3000 generations. Then we plot the final result of these 70 cities:

[2]The solution is not unique, which means there are other solutions that the route's total length is identical to the optimal route. e.g. $C \rightarrow B \rightarrow G \rightarrow D \rightarrow A \rightarrow E \rightarrow H \rightarrow F$

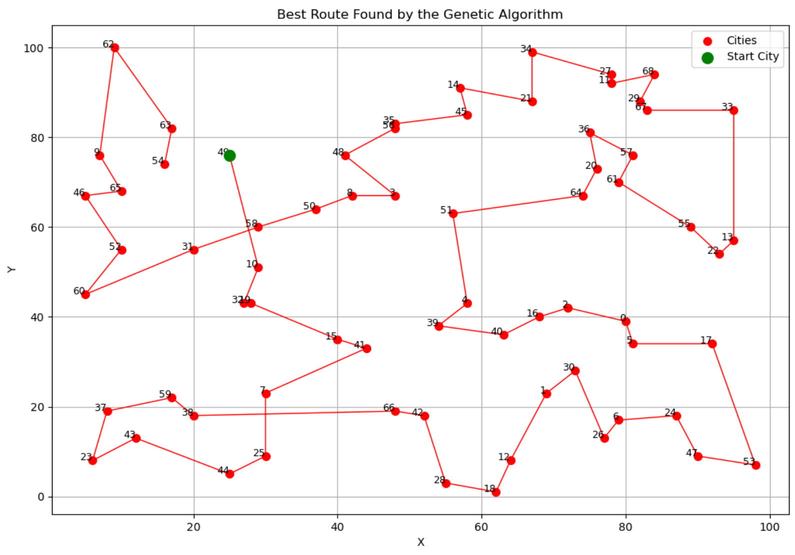

Figure 2: GA Best route

Where the Green Spot represent the start city, and Red Spot represent the others. We cancel the link of the tail city and start city to show the path more visually. The length of our computed optimal distance is 717.927.